Amendments to Specification:

Please update the paragraph beginning at line 13 through line 20, page 3, as follows:

Different methods have been proposed for carrying out modular multiplication. In particular, in an article appearing in "The Mathematics of Computation," Vol. 44, No. 170, April [[1995]] 1985, pp. 519-521, Peter L. Montgomery describes an algorithm for "Modular Multiplication without Trial Division." However, this article describes operations that are impractical to implement in hardware for a large value of *N*. Furthermore, the method described by Montgomery operates only in a single phase. In contrast, the system and method presented herein partitions operational cycles into two phases. From a hardware perspective, the partitioning provides a mechanism for hardware sharing which provides significant advantages.

Please update the paragraph beginning at line 17 through line 18, page 12, as follows:

Figure 3 is a block diagram similar to Figures 1 and 2 except more particularly showing those data flow paths which are active during the second or *Z*-phase of calculation[[.]];

Please update the paragraph beginning at line 4 through line 5, page 14, as follows:

Figure 13 (depicted as Figures 13A and 13B in the drawings) is a block diagram illustrating an improved rightmost processor element in which an adder in a critical path has been moved to improve performance;

Please update the paragraph beginning at line 15 through line 20, page 15, as follows:

The structure and operation of the present invention is dependent upon the partitioning of one of the multiplying factors into a plurality of *k* bit-wide pieces. Thus, instead of representing a binary number *A* as $\sum_{i=0}^{n-1} a_i 2^i$, one of the multiplying factors in the present invention is represented instead in the form $A_{m-1} R^{m-1} + \ldots + A_2 R^2 + A_1 R + A_0 = \sum_{j=0}^{m-1} A_j [[R^i]] \underline{R^j}$, where $R = 2^k$. In this representation, the number *A* is represented in block form where each of the *m* blocks includes *k* bits. That is, each A_i represents an integer having *k* bits.

Please update the abstract as follows:

The modular exponentiation function used in public key encryption and decryption systems is implemented in a standalone engine having at its core modular multiplication circuits which operate in two phases which share overlapping hardware structures. The partitioning of large arrays in the hardware structure, for multiplication and addition, into smaller structures results in a multiplier design ~~comprising~~ which includes a series of nearly identical processing elements linked together in a chained fashion. As a result of the two-phase operation and the chaining together of partitioned processing elements, the overall structure is operable in a pipelined fashion to improve throughput and speed. The chained processing elements are constructed so as to provide a partitionable chain with separate parts for processing factors of the modulus. In this mode, the system is particularly useful for exploiting characteristics of the Chinese Remainder Theorem to perform rapid exponentiation operations. A checksum mechanism is also provided to insure accurate operation without impacting speed and without significantly increasing complexity. While the present disclosure is directed to a complex system which includes a number of features, the present application is particularly directed to the incorporation and integration of circuits used for calculating a modular multiplicative inverse used as an input parameter to the process.